Amendments to the Specification:

Please replace the paragraph on page 78 line 16 to page 79 line 13 with the following amended paragraph:

a1

FIG. ~~29~~ 31 shows a preferred construction for the identity chip. The identity chip 350 includes encryption circuitry 351 and a "write-only" EEPROM memory 352 for storing at least one key 353. The key 353 can be written into the memory 352 from a data path 354 including external leads connected to the chip. The key 353 can be read from the memory 352 by the encryption circuitry, but the key cannot be read from the data path 354 or any other external leads connected to the chip. For example, the encryption circuitry ~~352~~ 351 includes a microprocessor 355 and a microcode read-only memory (ROM) 356 storing microcode executed by the microprocessor. The microprocessor is programmed to recognize a command from the data path 354 for writing into the memory 352 a key from the ~~buys~~ bus 354. The microprocessor is also programmed to recognize a command from the data path 354 for receiving a number from the data path, reading the key 353 from the memory 352, encrypting the number with the key, and transmitting the encryption result onto the data path. The microprocessor, however, will not recognize any command for transmitting the key onto the data path 354 or any other leads of the chip. In this sense, the memory 352 is a "write-only" memory. Moreover, the EEPROM memory 352 and at least the internal data path 357 to the memory 352 are covered by an upper layer of metal 358 (shown in dashed lines in FIG. 31) on the chip 350 so that it is virtually impossible for the key to be

recovered by probing, inspection, disassembly, or "reverse engineering" of the chip. The EEPROM 352 could store a plurality of different keys, and the microprocessor could recognize a command from the data path 354 for selecting which of the keys to use for encrypting the number received on the data path.

a1

Please replace the paragraph on page 79 line 15 to page 80 line 16 with the following amended paragraph:

FIG. 32 shows how identity chips 361, 363 are incorporated in the data processing system of FIG. 1 for authenticating host controller identity. In this example, the Fibre Channel loop 41 is used for access to highly sensitive data, such as configuration information, stored in the cached storage subsystem 20. The identity chip 361 in the host controller 61 stores a unique secret key 362 for the host controller 61, and the identity chip 363 in the host controller 62 stores a unique secret key 364 for the host controller 62. Copies of the keys 362, 364 are stored in a list 365 in the ~~host controller~~ port adapter 35. The list 365, for example, is a table, and each key in the list 365 is associated with the WWN of a respective host controller known to the port adapter. A copy of the list 365 of keys is stored in a logical storage volume of the storage subsystem, or the list of keys is stored in a nonvolatile portion of the port adapter memory 77, so that the list will be retained if power to the port adapter is lost. Such a nonvolatile portion of the port adapter memory 77 could be provided by one or more identity chips constructed as shown in FIG. ~~29~~ 31. The port adapter memory 77 further includes a list 366 of random numbers sent to

the hosts. The list 366, for example, is maintained as a queue. The list of random numbers 366 and the list of keys 365 are used by host authentication routines 367 in the port adapter microcode 79. As the random numbers are received by the hosts, the host controllers place the random numbers in similar lists 368, 269 in their respective memories 370, 371. As used in this specification, the term "random number" would include a so-called "pseudo-random number" so long as the number would appear to be selected at random during the typical duration of time that a host controller is logged in to a port adapter. For example, the random numbers transmitted by the port adapter to a host controller are generated by a conventional random number generator routine that is seeded at the time that the host controller logs in to the port adapter. The BASIC programming language, for example, provides such a random number generator routine.

a!